8.1-Naming Convention

In Java, it's considered acceptable practice to name classes, variables, and methods in a way that clearly describes their purpose. This approach, rather than using arbitrary names, helps improve the readability and maintainability of code. When naming these elements, Java developers typically follow CamelCase conventions, which involve capitalizing the first letter of each subsequent word in a compound word.

Advantages of Naming Conventions in Java

- Readability: Well-named identifiers make the code easier to read and understand for both the author and others who may use it later.

- Maintenance: Clear naming reduces the time required to understand the code, making it easier to update and maintain.

CamelCase Naming Conventions

- Classes and Interfaces: The first letter is uppercase. For example: Thread, Runnable, Reflection, Collections.

- Methods: Start with a lowercase letter, with subsequent words capitalized. For example: concat(), toString(), wait(), notify().

- Constants: Written in all uppercase letters, often with words separated by underscores. For example: PIE, SUNDAY, MAX_VALUE.

TELUSKO

Popular Naming Conventions

Identifier Type	Naming Rules	Examples
Class	- Start with an uppercase letter. - Should be a noun. - Use meaningful words and avoid acronyms.	`public class Employee { }`
Interface	- Start with an uppercase letter. - Should be an adjective. - Avoid acronyms.	`interface Printable { }`
Method	- Start with a lowercase letter. - Should be a verb. - Use camelCase for multi-word names.	`void draw() { }`
Variable	- Start with a lowercase letter. - Avoid special characters and one-character names. - Use camelCase for multi-word names.	`int id;`
Package	- Use all lowercase letters. - Separate multi-word names with dots.	`package com.telusko;`
Constant	- Use all uppercase letters. - Separate multi-word names with underscores. - May contain digits but not as the first character.	`static final int MIN_AGE = 18;`